Continuous Integration (CI) and GitLab CI

Petr Šťavík
Why CI

- During a development process, multiple issues with integration can occur:
  - branch diverged too much from its source
  - some developer forgot to run local build
  - some developer forgot to run local tests
  - conflicts, ...

- CI tries to prevent these integration issues
What is CI

- by a "textbook" definition, it is a practice, where developers try to integrate more often (at least once per day) and in smaller chunks
- to enable this frequent integration, it is useful to setup
  - automatic build process
  - automatic test process
- typically combined with so called Continuous Delivery - frequent releases, then denoted as "CI/CD" (typically automatic as well)
CI in practice

- in practice (from my experience), using CI means having a following setup:
  - some versioned repository
  - CI server hooked to this repository

  - it detects new pushes (by e.g. polling) and it executes predefined (configurable) pipeline, like:
    - build (+ report about result)
    - test (+ report about result)
    - release (if we, e.g. merge with master branch)
    - if any of these stages fail, pipeline stops and error is reported
Tools for CI support

- Providers offer their implementation of CI server and runners that run individual stages of pipeline.
- CI server typically configurable via some web UI.

**Agents**

<table>
<thead>
<tr>
<th>Agent</th>
<th>Status</th>
<th>Running build</th>
<th>Stop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux Agent</td>
<td></td>
<td>TFS 2015 :: Build</td>
<td>Running</td>
</tr>
<tr>
<td>Mac OS Agent</td>
<td></td>
<td>TFS 2015 :: Build</td>
<td>Running</td>
</tr>
<tr>
<td>Windows Agent</td>
<td></td>
<td>TFS 2015 :: Build</td>
<td>Running</td>
</tr>
</tbody>
</table>
Tools for CI support - cont.

- runner configuration - name, CI server, token for authorization
- runner count is a typical restriction of free versions (TeamCity - 3 runners)
- providers sometimes offer infrastructure for CI server and runners for free (then limited by e.g. cpu time)
GitLab

- alternative option for Github, BitBucket, ..
- better support for free version
  - private repositories
  - issues, milestones, wiki
  - CI support
GitLab CI

- integrated into gitlab
  - you don't have to setup CI server separately and connect it with your repository
  - runners are of two types
    - basic ones - running on your own machines
    - shared runners - running of infrastructure of gitlab (even with free version, then capped at 2000 minutes per project)
- it is very easy to setup - you just have to define the pipeline
- one downside is, that you can wait a bit due to requests from other users
.gitlab-ci.yml

- file describing the pipeline
- you need to place it in top-level directory of your repo
  - and that's it!! with that, you have the CI set up
- pipeline is divided into stages
- within each stage, several jobs can be performed
- each job is run independently
- job's body is a shell script defined in the script clause

```yaml
stages:
  - build
  - test
  - deploy

build_job_1:
  stage: build
  script:
    - echo "inside build_job_1"

build_job_2:
  stage: build
  script:
    - echo "inside build_job_2"

test_job_1:
  stage: test
  script:
    - echo "inside test_job_1"

test_job_2:
  stage: test
  script:
    - echo "inside test_job_2"

deploy_job_1:
  stage: deploy
  script:
    - echo "inside deploy_job_1"
```
Demos time

- (simple demo)
- (demo with failed compilation and failed tests)
- (demo with deploy and better deploy)
- other options in interface (schedules, charts, envrionments like production, testing, ...)

Other options for .gitlab-ci.yml file

- only and except can limit when a new job is created
  - for e.g. tags, specific branches, scheduled pipelines
- when
  - possible values - on_success, on_failure, always, manual
- cache
  - files may be cached between jobs (that are otherwise independent)
- retry (specify retry count if a job fails), include (include external .yaml files like templates), variables, ...
Using your own runners

- steps are pretty simple
  - download a runner
  - create token for project
  - register runner